REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) NPRDC TR 87-35	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Navy Personnel Research and Development Center 6b. OFFICE SYMBOL (*If applicable*) Code 61	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (*City, State, and ZIP Code*) San Diego, CA 92152-6800	7b. ADDRESS (*City, State, and ZIP Code*)
8a. NAME OF FUNDING/SPONSORING Office of the Chief of Naval Research 8b. OFFICE SYMBOL (*If applicable*)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (*City, State, and ZIP Code*) Arlington, VA 22203	10. SOURCE OF FUNDING NUMBERS PROGRAM: 61152N, 62763N; PROJECT: RR000, RR63-521; TASK: 01-042, 804-031; WORK UNIT: 04-025, 03.06

11. TITLE (*Include Security Classification*)
An Alternative Algorithm for Optimizing Personnel Assignment in the Navy

12. PERSONAL AUTHOR(S)
Krass, Iosif A.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (*Year, Month, Day*)	15. PAGE COUNT
Final	FROM 86 Jan TO 86 Jul	1987 September	33

16. SUPPLEMENTARY NOTATION
See NPRDC TRs 84-49 and 84-52 for descriptions of related work.

17. COSATI CODES			18. SUBJECT TERMS (*Continue on reverse if necessary and identify by block number*)
FIELD	GROUP	SUB-GROUP	
05	09		Optimization, assignment

19. ABSTRACT (*Continue on reverse if necessary and identify by block number*)

This report describes an alternative to the network optimization computer algorithm to solve the Navy's assignment problem, using a heuristic "search-by-chain" approach. This algorithm will prove useful as a check on the quality of solutions to operational models being implemented using a network with side constraints algorithm, which also incorporates heuristic techniques.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Krass, Iosif A.	22b. TELEPHONE (*Include Area Code*) (619) 225-2371 22c. OFFICE SYMBOL Code 61

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

Navy Personnel Research and Development Center
San Diego, CA 92152-6800 NPRDC-TR 87-35 September 1987

An Alternative Algorithm for Optimizing Personnel Assignment in the Navy

NPRDC TR 87-35

September 1987

An Alternative Algorithm for Optimizing Personnel Assignment in the Navy

Iosif A. Krass, Ph.D.

Reviewed by
Tom Blanco

Approved by
Joe Silverman

Released by
B. E. Bacon
Captain, U.S. Navy
Commanding Officer

and

J. S. McMichael, Ph.D.
Technical Director

Navy Personnel Research and Development Center
San Diego, California 92152-6800

FOREWORD

This research was conducted within independent research subproject RR000-01-042-04-025 (Approaches to Multiple Objective Assignment), under the mission sponsorship of the Navy Personnel Research and Development Center, and exploratory development subproject RF-63-521-804-031 (Career Management Planning), under the mission sponsorship of the Office of the Chief of Naval Research. The objective of these projects is to develop new technologies to improve the Navy's personnel assignment system.

This report is the third in a series resulting from these projects. Previous reports described a network optimization approach to solve the Navy's personnel assignment problem (NPRDC TR 84-49) and a network model for use in estimating permanent change of station (PCS) costs incurred in the assignment process (NPRDC TR 84-52). This report describes an alternative computer algorithm for solving the Navy's assignment problem in particular, and any assignment problem with a many side constraint, in general.

The heuristic "search-by-chain" algorithm described will prove useful as a check on the quality of solutions to the Navy's assignment problem that are based on a network with side constraints. These solutions are also obtained by heuristic procedures. This report should be of interest to researchers working on large-scale optimization problems that require integer solutions.

B. E. BACON
Captain, U.S. Navy
Commanding Officer

JAMES McMICHAEL
Technical Director

SUMMARY

Problem

Nonstandard, or "side," constraints make optimal solutions to the Navy's personnel assignment problem difficult to compute. These constraints can vary from the balance of personnel assets among fleets to school quotas for training en route to a new assignment. Recently, a capacitated transshipment model formulation and computationally efficient network flow algorithm solved part of the Navy's assignment problem. In addition, in order to solve more complicated assignment problems, we are modifying a general network algorithm to incorporate some types of side constraints. However, the algorithm is limited in its ability to consistently produce optimal integer solutions.

Objective

The objective of this effort was to develop an alternative approach and computer algorithm for solving the Navy's assignment problem, in particular, and an assignment problem with a large number of side constraints, in general.

Approach

Using the "penalty relaxation" method, we converted the assignment problem with side constraints to a classical assignment problem with a nonlinear objective function. This approach consisted of two steps. The first step generates an initial solution that is close to optimal by using an approach developed in Kuhn's (1953) Hungarian assignment algorithm. The second step improves the initial solution by modifying the search-by-tree to a method of searching among feasible solutions by chain.

Results and Discussion

For the Navy's assignment problem, formulated as a capacitated transshipment model, the presented algorithm, written in FORTRAN, takes 5 percent to 30 percent more computing time than the

computationally efficient network code. However, the algorithm has much more flexibility and the potential to incorporate more side constraints, and therefore to handle more complex Navy assignment problems.

Recommendation

Future development should concentrate on making the algorithm more computationally efficient by using more advanced computer languages, such as assembly and C. The algorithm should be developed concurrently with and as an alternative to network approach.

Table of Contents

List Of Tables

INTRODUCTION

Background

Matching people to jobs in the Navy is a large combinatorial problem that is further complicated by nonstandard, or "side," constraints. One set of constraints concerns allocation goals for balancing personnel assets among fleets, priority jobs, sea and shore duty, and male only, female only, and mixed-gender jobs. Other side constraints include advanced training school quotas for training en route to a specialized skill assignment and permanent change of station budget restrictions. These constraints make the Navy assignment problem less amenable to standard optimization techniques. Early attempts to solve the problem used classical linear or special linear integer programming codes. These codes were hampered by limitations on the amount of constraints, the amount of computer processing time, and the unreliability of integer solutions (i.e., in assigning a whole person to a whole job).

Problem

Commercial codes are available for solving the assignment problem using the network approach (Glover, Karney, Klingman & Russel, 1978; Jensen & Barnes, 1980; Glover, Klingman & Ross 1973b; Karney & Klingman, 1976; Mulvey, 1978) or the transportation approach (Charnes & Cooper, 1961; Glover, Klingman & Russel, 1973a). In both approaches, an assignment problem is embedded in a more general class of special integer linear programming problems, for which well known computational methods exist. Unfortunately, these approaches do not work when an assignment problem contains side constraints.

Recently, a capacitated transshipment model formulation (Liang, 1984; Liang & Lee, 1985) and computationally efficient network flow algorithms (Bradley, Brown & Graves, 1977; Kennington & Helgason, 1980) solved part of the Navy's assignment problem by heuristically incorporating all allocation goals as transshipment arcs and avoiding the creation of side constraints. In addition, these models modify the general network algorithm to incorporate school quotas for en route training. However, algorithms of this type (Barr, Farhangian & Kennington, 1986; Chen & Saigal 1977) require further

development to consistently obtain optimal integer solutions. Other approaches need to be investigated to solve the Navy's assignment problem.

Objective

The objective of this effort was to develop an alternative approach and computer algorithm for solving the Navy's assignment problem and, in general, an assignment problem with a large number of side constraints.

MODEL FORMULATION

Classical Assignment Problem

The classical assignment problem consists of choosing a position for every one of n persons among n opportunities under the condition of minimizing total cost of all assignments. For every person i and job j, there is a cost c_{ij} of assigning the person i to the job j. An optimal assignment should provide a minimum total cost for n persons.

In mathematical terms, the problem is

$$\min \sum_{i,j} c_{ij} x_{ij}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad j = 1,...,n \tag{1}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad i = 1,...,n, \tag{2}$$

where $x_{ij} \in \{1,0\}$. When person i is assigned to job j, $x_{ij} = 1$; otherwise $x_{ij} = 0$. Equations (1) and (2) insure that one person is assigned to each job and one job is assigned to each person, respectively.

The above problem is an integer linear programming problem with a unimodular matrix. The unimodularity of the constraint matrix of a linear programming problem means that any basic feasible solution of the problem is integer. Special algorithms have been built to efficiently solve this problem, as mentioned earlier in this report.

Assignment Problem With Side Constraints

A good example of an assignment problem with side constraints is the Navy assignment problem. The side constraints of the Navy assignment problem may cause the constraint matrix to lose its unimodularity property, so the special assignment or transportation algorithms should not be used. The side constraints for setting allocation goals for balancing personal assets among various dimensions were first documented in Blanco, Liang, Habel, and Ritter, 1984 and developed in Liang and Thompson, 1986. Rather than formulating this complex problem as a capacitated transshipment model and using a network code as documented previously, we converted this problem to an assignment problem with a nonlinear objective function using a "penalty relaxation" method.

In the case of the Navy assignment problem, all jobs are divided into k major groups, depending on some characteristics, or parameters, of the job (billet). These parameters include gender requirements, duty requirements and general type of job. Some of the major groups should be filled at least with a minimal manning percentage; others should be filled at most with a maximal manning percentage. In every major group, there is an additional splitting of jobs into l subgroups, defined by different fleets (we will assume that $l = 3$). The manning percentage of different subgroups belonging to the same major group should be balanced.

As a rule, the amount of available jobs m is more than the amount of available people n; and not all people are eligible for every job. Without loss of generality, we assume that every person is eligible for every job. In the case of assigning a person to a job for which he or she is not really eligible, we give the cost c_{ij} of this assignment as M, where M is a rather big number (penalty) for wrong assignment.

Let S_r be the set of all jobs for a major group r, where

$$S_r \subset \{1,...,m\}, \quad r = 1,...,k$$

$$\bigcup_{r=1}^{r=k} S_r = \{1,...,m\}.$$

Let I_1 be the set of groups that need to be filled with a minimal manning percentage, where $I_1 \subset \{1,...,k\}$. Then for $r \in I_1$ we have constraints:

$$\frac{1}{V_r}(P_r + \sum_i \sum_{j \in S_r} x_{ij}) \geq 1, \tag{3}$$

where V_r is the amount of billets authorized for group r; and P_r is the current manning of the group. Here and below we assume that in summation, index i goes from 1 to n if not stated directly otherwise.

In the same way, if I_2 is the set of groups that need to be manned at not more than 100 percent and $r \in I_2$, then

$$\frac{1}{V_r}(P_r + \sum_i \sum_{j \in S_r} x_{ij}) \leq 1. \tag{4}$$

There are two meta groups of jobs that also have minimal manning percentage constraints. These groups are jobs to be filled by males or females only. If J_1, J_2 are sets of jobs required to be filled by males and females respectively, we can write an analog of constraint (3) for this group, as follows:

$$\frac{1}{W_k}(U_k + \sum_i \sum_{j \in J_k} x_{ij}) \geq 1, \quad k = 1,2. \tag{5}$$

Here W_k is the number of billets authorized in the group J_k ($k = 1,2$), and U_k is the current manning of the group.

To work with equal manning percentage requirements between subgroups of the same major group, let us denote S_{rl} as the set of jobs in the group r for fleet l ($l = 1,2,3$). Obviously $S_{r1} \cup S_{r2} \cup S_{r3} = S_r$; $r=1,...,k$. Let

$$M_r = \max_{l=1,2,3} \frac{1}{V_{rl}}(\sum_i \sum_{j \in S_{rl}} x_{ij} + P_{rl}), \qquad r = 1,...,k. \tag{6}$$

In this equation, V_{rl} is the number of billets authorized for the rl group, and P_{rl} is the current manning of the group.

Also denote:

$$m_r = \min_{l=1,2,3} \frac{1}{V_{rl}}(\sum_i \sum_{j \in S_{rl}} x_{ij} + P_{rl}), \qquad r = 1,...,k. \tag{7}$$

Then our problem is to minimize the distance $M_r - m_r$. Let x^+ be a convex function:

$$x^+ = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

Putting together constraints (3) through (7), we can write the objective function:

$$Q = \sum_{i,j} c_{ij}x_{ij} + q_1 \times \sum_{k=1}^{k=2} (W_k - U_k - \sum_i \sum_{j \in J_k} x_{ij})^+ + q_2 \times \sum_{r \in I_1} (V_r - P_r - \sum_i \sum_{j \in S_r} x_{ij})^+ + \tag{8}$$

$$+ q_3 \times \sum_{r \in I_2} (\sum_i \sum_{j \in S_r} x_{ij} - V_r + P_r)^+ + q_4 \times \sum_{r=1}^{r=12} (M_r - m_r).$$

This should be minimized under the constraints (1) and (2). Here q_1, q_2, q_3 and q_4 are penalty coefficients for not meeting corresponding constraints. Because we have more jobs than people, instead of the constraint (2) we should have

$$\sum_{i=1}^{n} x_{ij} \le 1; \quad j = 1,...,m. \tag{2a}$$

Note that substituting of an inequality (2a) for the equality (2) is not very essential because we always can add $m - n$ new fictitious persons, which will give zero input in the objective function (8) (i.e. $c_{ij} = 0$ for them). Therefore we will not change the main problem but return to equality (2).

SOLUTION TECHNIQUES

Hungarian Method to Solve the Classical Assignment Problem

Kuhn's 1953 approach to solve the classical assignment problem is used for getting an initial solution in our algorithm. Kuhn's method is a primal-dual method; we describe the dual problem for the original assignment problem shown in (1),(2). The dual problem is

$$\max \sum_{i=1}^{n} u_i + \sum_{j=1}^{n} v_j,$$

subject to

$$u_i + v_j \le c_{ij}; \quad i,j=1,n, \tag{10}$$

where u_i and v_j are dual variables that are to some extent potential weights of the importance of the i-th person or the j-th job for assignment. From the theory of duality, it is known that if we find a set of potentials $\{u_i\}$, $\{v_j\}$ satisfying inequality (10) and a set of $\{x_{ij}\}$ satisfying equalities (1),(2) that relate to each other by the equality

$$(c_{ij} - u_i - v_j) \times x_{ij} = 0; \quad i,j=1,..n, \tag{11}$$

then u_i, v_j, x_{ij} are an optimal solution for both the dual and original problems, respectively. The

equality (11) is a complementary slackness condition. There is one feasible solution satisfying the dual constraints (10):

$$\bar{u}_i = \min_j \{c_{ij}\} \tag{12}$$

$$\bar{v}_j = \min_i \{c_{ij} - \bar{u}_i\}. \tag{13}$$

In other words first we calculate the minimum by row to get $\bar{u}_i$ in equality (12) and then, subtracting these minimums from corresponding rows of the cost matrix and taking the minimum by column, we get $\bar{v}_j$. This process, together with checking the complementary slackness conditions (11), gives rise to the Hungarian algorithm. In this algorithm, beginning from the original cost matrix $\{c_{ij}\}$, we deduce a reduced cost matrix $\{\tilde{c}_{ij}\}$ whose elements are calculated as follows:

From every element of a row, we will subtract the minimum element of this row; and then from every element of a column, we will subtract minimum element of this column. This process corresponds to the equalities (12),(13) for creating a feasible dual solution. After this process we will get a reduced matrix $\{\tilde{c}_{ij}\}$ that contains many zero elements. Zero elements constitute a covering set if in every row and in every column there is at least one zero; and in any submatrix that is organized by taking off the row and the column corresponding to an element of the covering set, the subset of the covering set is also a covering set. Choosing $x_{ij} = 1$ in the places of those zeroes that constitute a covering set, we satisfy the equalities (11),(1),(2); and we get an optimal solution. However, if the reduced matrix does not have this covering set of zeroes, there is a process for successive matrix reduction, which is described in Bazaraa and Jarvis (1977). The case of no covering set of zeroes is much more complicated than the original process (12),(13). The reduction process mentioned above requires the application of network optimization or some another technique, which is a big drawback for the original Hungarian method.

One difficulty in this method occurs when the reduced matrix has more than one zero in a row: which one should be used in the covering set? Analysis of the algorithm shows that this decision depends on non-zero values in the corresponding columns of the reduced matrix. In most cases a zero element in the reduced matrix will correspond to solution $x_{ij} = 1$ if $\min_{i \in S_1} \tilde{c}_{ij}$ is bigger, where S_1 is the set of non-zero elements in the row j. This property leads to the gradient approach for getting an initial

solution. In other words, the method of finding an initial solution consists of finding a person with maximal distance between "minimizing job" and "next to minimizing job" for him, where we measure distance in the sense of difference between job costs.

Gradient Approach to Initial Solution

Let us call a gradient for a person i in the job j_2 with respect to the job j_1 the difference $c_{ij_2} - c_{ij_1}$. Then the gradient method of finding an initial solution consists first of finding for a person i a job that provides a minimal but not a zero gradient with respect to minimizing job j. Then we find the maximum of all these gradients among all persons and assign this person to a minimizing job. Next we describe this process more exactly and calculate the complexity of this part of the algorithm, in the sense of the amount of elementary operations necessary for getting an initial solution.

For each person i, denote as J_i^n the set of jobs j_1 such that

$$c_{ij_1} \le c_{ij}; \qquad j \in \{1,...,n\}. \tag{14}$$

In other words, J_i^n is the set of minimizing jobs for a person i. Define

$$\Delta_n(i) = \min_{j \notin J_i^n}(c_{ij} - c_{ij_1}); \qquad j_1 \in J_i^n. \tag{15}$$

$\Delta_n(i)$ is the gradient for a person i with respect to a minimizing job. If $n \neq 1$ and jobs are of varying costs, then gradient $\Delta_n(i) > 0$, because in equality (15) we have $j \notin J_i^n$. Finally we define a set I_n as a set of i_1 such that

$$\Delta_n(i_1) \ge \Delta_n(i); \qquad i \in \{1,...,n\}. \tag{16}$$

In other words, I_n consists of persons who maximize their minimal gradient, as defined in equality (15).

Calculating the initial solution consists of assigning to a person $i_1 \in I_n$ a job $j_1 \in J_{i_1}^n$. Then after subtracting person i_1 and job j_1 out of sets of all available persons and jobs, we repeat this process until we get an assignment for each person. If we denote as d a constant for estimating computing time for comparison and subtraction, then following inequality (14), the calculation of $j_1 \in J_i^n$ will take not more than $n \times d$ time. The calculation of $\Delta_n(i)$ will require not more than $n^2 \times d$ time, and calculation of $i_1 \in I_n$ will require not more than $n^3 \times d$ time. The whole calculation of an initial solution will take

not more than

$$(n^3 + (n-1)^3 + (n-2)^3 + \ldots) \times d \leq C \times n^4 \qquad (17)$$

computer time, where C is a constant. The estimation (17) can be considerably improved in practical calculations if we save a list of minimizing jobs in (14) and (15) for every person for the next round of calculations.

If a reduced matrix in the Kuhn method has only one zero in every row and in every column, then the above initial solution will coincide with an optimal; but there are also other cases when this occurs.

Let $j_1 \in J_{i_1}^n$ and $J_{n-1} = \{1,\ldots,n\} - \{j_1\}$. That is, J_{n-1} is set of all jobs except j_1, which is already used for a person i. As in inequality (14) let J_i^{n-1} denote a set of all jobs j_2 such that

$$c_{ij_2} \leq c_{ij} \qquad j \in J_{n-1} \qquad (18)$$

We call a cost matrix monotone at step 1 if $J_i^{n-1} \subseteq J_i^n$ for all i. If a matrix is monotone at step 1, then taking person i_1 and the job j_1 out of the available sets of persons and jobs, respectively we can define a matrix as monotone at step 2 by analog with the previous definition. The cost matrix is fully monotone if it is monotone at all $(n-1)$ steps. An example of a fully monotone matrix is

$$\begin{bmatrix} 4 & 4 & 4 & 4 \\ 8 & 3 & 3 & 3 \\ 4 & 8 & 2 & 2 \\ 7 & 6 & 5 & 1 \end{bmatrix}.$$

Really, if the first person will choose the first job that belongs to J_1^4, it does not affect the amount of minimizing jobs for other persons. The same consideration is true for other persons. In the above case, the reduced matrix has more than one zero in a row.

Lemma 1. If the cost matrix of an assignment problem is fully monotone, the initial solution coincides with an optimal solution.

(The proof of this lemma as well as others is in the appendix).

There is another case of coincidence of an initial solution with an optimal solution.

Let us consider

$$\delta_n(i) = \max_{j, j_1 \in \{1,\ldots,n\}} (c_{ij} - c_{ij_1}); \qquad j_1 \in J_i^n. \tag{19}$$

That is, $\delta_n(i)$ is the maximum of all distances between costs for a person i. Let

$$\delta_k(i) = \max_{j, j_1 \in \{1,\ldots,n\} - \{\hat{j}_1, \ldots, \hat{j}_{k-1}\}} (c_{ij} - c_{ij_1}), \tag{20}$$

where $\hat{j}_k$ is the job assigned to person i_k at step k of the initial solution algorithm. In equality (20) we assume that $i \in \{1,\ldots,n\} - \{i_1,\ldots,i_{k-1}\}$, that is, person i is not assigned yet. The value $\delta_k(i)$ is analogous to $\delta_n(i)$ in (19) at the step k of the initial algorithm.

Lemma 2. Let $J_{i_k}^k$ be one element set, and let

$$\Delta_{n-k+1}(i_k) \geq \delta_{n-k+1}(i)$$

for $i \in \{1,\ldots,n\} - \{i_1,\ldots,i_k\}$, then an initial solution coincides with an optimal solution.

An example of this kind of cost matrix is:

$$\begin{bmatrix} 1 & 11 & 21 \\ 12 & 17 & 22 \\ 3 & 5 & 7 \end{bmatrix}.$$

Really, in this case $i_1 = 1$. We have $\Delta_3(1) = 10$, and $\delta_3(2) = 10$; $\delta_3(3) = 4$; $\Delta_2(2) = 5$, and $\delta_2(3) = 4$. In other words, on every step of the algorithm, a gradient for the assigned person is rather big compared with all gradients in the remaining part of cost matrix.

The described cases show that the initial solution calculated by the gradient approach is sometimes an optimal solution. In addition, the gradient approach provides the assignment of a minimizing job to a person on every step of the algorithm in such a way that substitution of this assignment to another will be blocked by getting an additional cost that is equal to the correspondent gradient. By choosing a maximization of the gradient as a rule for initial assignment, we can expect closeness of the initial assignment to an optimal solution. In all our testing, we do get this expected closeness.

Using a more complicated initial solution algorithm, we can get more cases where the initial solution coincides with an optimal solution. However, we found that it is not worthwhile, because it will lead to considerable increase in computational time without much compensation in the second part of the algorithm.

The described initial solution algorithm can be easily adjusted for the assignment problem with side constraints by using penalty relaxation. To do this we have to recalculate costs in formulas (14) - (15), corresponding with a new objective function. Due to the nonlinearity of the objective function, this recalculation should be done during calculations of corresponding minimums and maximums.

Searching by Improving Chains

For cases where initial solutions calculated by the gradient approach do not coincide with an optimal solution, we developed a second algorithm for improving the initial solution. Below we describe this "improving part."

Any assignment can be expressed as a vector $\{j_1,\ldots j_k,\ldots j_n\}$, where j_k is a job assigned to person k. Two assignments $\{j_1^1,\ldots j_k^1,\ldots j_r^1,\ldots j_n^1\}$ and $\{j_1^2,\ldots j_n^2\}$ differ on elementary permutation if $\{j_1^2,\ldots j_n^2\} = \{j_1^1,\ldots j_r^1,\ldots j_k^1,\ldots j_n^1\}$. In other words, in the assignment $\{j_1^2,\ldots j_n^2\}$, persons r and k changed their jobs with respect to original assignment $\{j_1^1,\ldots j_n^1\}$. To return from assignment $\{j_1^2,\ldots j_n^2\}$ person r should change jobs with person k; in other words, there is a chain of length one connecting these two assignments. We denote the elementary permutation that corresponds to the described above chain of length one as (r,k). If the assignment $\{j_1^2,\ldots j_n^2\}$ differs from the original assignment $\{j_1^1,\ldots,j_n^1\}$ in two elementary permutations $(k,s),(s,r)$, then we say that these assignments differ in a chain of length two. In this case, to restore the original assignment from the new $\{j_1^2,\ldots j_n^2\}$, requires at least three different persons. Chains of length three, four and so on connecting two given assignments can be defined in the same way.

Consider an assignment $\{j_1^1,\ldots j_n^1\}$. We say that there is an improving chain of length k if there is an assignment $\{j_1^2,\ldots j_n^2\}$ with a cost less than the original assignment $\{j_1^1,\ldots j_n^1\}$ and that connects with the original assignment by a chain of length k. Any two assignments can be connected by chains of elementary permutations.

Lemma 3. If the assignment $\{j_1,\ldots j_n\}$ is not optimal, there is an improving chain of length $k \le n$.

We will present two examples of the application of searching by chains in the next section.

To find all improving chains of length k, we should check C_n^k possible chains, which gives an exponential estimation of effectiveness for an optimizing algorithm based on Lemma 3. However, this estimation can be considerably improved based on the following consideration:

Lemma 4. If for assignment $\{j_1,...,j_n\}$ there is an improving chain of length one, then there is an elementary permutation (r,k) such that $c_{rj_r} > c_{rj_k}$.

In other words, under the lemma's condition for a person r, there is a job j_k that differs from the job j_r assigned to him and that costs less. The same kind of property exists for improving chains with lengths of more than one.

If the cost matrix satisfies some additional properties, the amount of time for the chain part of the algorithm can be cut further. But even a simple property like Lemma 4, together with a good initial solution, will dramatically help decrease computational time. We found that stopping after checking improving chains of length one and two almost always gives an optimal solution.

As in the case of the initial solution part of the algorithm, this part can be easily applied for penalty relaxation of an assignment problem with side constraints. Again, due to non-linearity of the objective function, correction of costs should be done at the time of checking for corresponding improving chains.

NUMERICAL EXAMPLES

In this section we will demonstrate the work of the algorithm first without side constraints and then with side constraints. The example will be for the cost matrix:

$$\begin{bmatrix} 12 & 16 & 20 & 16 \\ 9 & 12 & 15 & 16 \\ 6 & 8 & 10 & 16 \\ 3 & 4 & 5 & 9 \end{bmatrix}. \qquad (21)$$

The calculation of an initial solution takes three steps.

At Step 1 from the cost matrix (21) we have

$$12 = \min_{j=1,2,3,4} c_{1j}; \qquad \Delta_1^{(1)} = 4;$$

$$9 = \min_{j=1,2,3,4} c_{2j}; \qquad \Delta_2^{(1)} = 3;$$

$$6 = \min_{j=1,2,3,4} c_{3j}; \qquad \Delta_3^{(1)} = 2;$$

$$3 = \min_{j=1,2,3,4} c_{4j}; \qquad \Delta_4^{(1)} = 1.$$

Here $\Delta_i^{(k)}$ is the *maxmin* gradient, which is defined in the gradient approach process for a person $i = 1,2,3,4$ at step k. Due to the definition of the gradient approach, we will get $j_1 = 1$ and the new cost matrix for assigning three persons to three jobs. The new matrix is part of the original matrix:

$$\begin{bmatrix} \cancel{12} & \cancel{16} & \cancel{20} & \cancel{16} \\ 9 & 12 & 15 & 16 \\ 6 & 8 & 10 & 16 \\ 3 & 4 & 5 & 9 \end{bmatrix}.$$

Here we crossed out elements that do not participate in the next steps.

At Step 2 we have

$$12 = \min_{j=2,3,4} c_{2j}; \qquad \Delta_2^{(2)} = 3;$$

$$8 = \min_{j=2,3,4} c_{3j}; \qquad \Delta_3^{(2)} = 2;$$

$$4 = \min_{j=2,3,4} c_{4j}; \qquad \Delta_4^{(2)} = 1.$$

From this we will get $j_2 = 2$ and the new cost matrix (for two persons and two jobs) which is a part of the original matrix:

$$\begin{bmatrix} \cancel{12} & \cancel{16} & \cancel{20} & \cancel{16} \\ 9 & \cancel{12} & \cancel{15} & \cancel{16} \\ 6 & \cancel{8} & 10 & 16 \\ 3 & \cancel{4} & 5 & 9 \end{bmatrix}.$$

At Step 3:

$$10 = \min_{j=3,4} c_{3j}; \qquad \Delta_3^{(3)} = 6;$$

$$5 = \min_{j=3,4} c_{4j}; \qquad \Delta_4^{(3)} = 4;$$

and $j_3 = 3$, $j_4 = 4$. Finally, the initial solution is $j_1 = 1$, $j_2 = 2$, $j_3 = 3$, $j_4 = 4$; and we are ready for the improving part of the algorithm.

We begin with searching chains of length one. For person 1 we have a minimizing assignment; so due to Lemma 4, there are no improving chains. Again due to Lemma 4, for person 2 there is only one "suspicious" element, c_{21}, because $9 = c_{21} < c_{22} = c_{2j_2} = 12$. Because the first job due to the initial solution is assigned to person 1, the corresponding chain to be checked for improving the assignment is (2,1). But for this chain we have

$$c_{11} + c_{22} = c_{1j_1} + c_{2j_2} = 12 + 12 < c_{21} + c_{12} = 9 + 16.$$

Therefore this chain is not improving. For person 3 there are two suspicious elements, c_{31} and c_{32}, which give rise to two chains (3,1) and (3,2) which are not improving. At last, for person 4 there are three suspicious elements, c_{41}, c_{42}, and c_{43}, which yield three chains (4,1), (4,2) and (4,3), from which the first and the second are improving.

$$c_{11} + c_{44} = c_{1j_1} + c_{4j_4} = 12 + 9 > c_{41} + c_{14} = 3 + 16.$$
$$c_{22} + c_{44} = c_{2j_2} + c_{4j_4} = 12 + 9 > c_{42} + c_{24} = 4 + 16.$$

Using the improving chain (4,1), we will get a new solution: solution $j_1 = 4$; $j_2 = 2$; $j_3 = 3$; $j_4 = 1$. Beginning from this solution we have for the first person a suspicious chain (1,4); for the second person, the suspicious chain (2,4); and for the third person, two suspicious chains, (3,4) and (3,2). The chain (2,4) for person 2 is improving, and using this we get a new solution $j_1 = 4$; $j_2 = 1$; $j_3 = 3$; $j_4 = 2$. After that we can find that the chain (3,4) for person 3 is improving, and using this chain we get the solution: $j_1 = 4$, $j_2 = 1$, $j_3 = 2$, $j_4 = 3$. Because there aren't more improving chains, this solution is an optimal solution.

To demonstrate the algorithm for the side constraints case, let us put two additional constraints on the original assignment problem with cost matrix (21) :

$$x_{11} + x_{22} + x_{33} + x_{44} \le 2,$$
$$x_{21} + x_{32} + x_{43} \le 2.$$

The penalty relaxation of this problem will be

$$L = \sum_{i,j} c_{ij}x_{ij} + q_1(x_{11} + x_{22} + x_{33} + x_{44} - 2)^+ +$$
$$+ q_2(x_{21} + x_{32} + x_{43} - 2)^+.$$

Let us assume $q_1 = q_2 = \frac{1}{2}\max_{i,j} c_{ij} = 10$ and apply the algorithm for this problem. Note that the side

constraints are made in such a way as to exclude the original initial solution as well as the original optimal solution.

The first two steps of the initial solution part of the algorithm will go without any effects of added side constraints. Then at Step 3 we will have

$$16 = \min\{c_{33} + 10, c_{34}\}; \qquad \Delta_3^{(3)} = 4;$$
$$5 = \min\{c_{43}, c_{44} + 10\}; \qquad \Delta_4^{(3)} = 14.$$

Thus we will get the initial solution $j_1 = 1$; $j_2 = 2$; $j_3 = 4$; $j_4 = 3$. In the improving part of the algorithm, there is an improving chain (1,3) of length one for person 3, which will lead to the solution $j_1 = 4$, $j_2 = 2$, $j_3 = 1$, $j_4 = 3$. The suspicious chain (2,3) for the second person, which should be improving without side constraints, now is not improving, due to the second term in the penalty objective function. Because there are no more improving chains, this solution is optimal.

Now we present an example that is a slight modification of example (21), which has no improving chain of length one but has an improving chain of length two.

$$\begin{bmatrix} \underline{12} & 16 & 20 & \overline{19} \\ \overline{9} & \underline{12} & 15 & 16 \\ 6 & \overline{8} & \underline{10} & 16 \\ 3 & 4 & \overline{5} & \underline{9} \end{bmatrix}. \tag{22}$$

As in the case of cost matrix (21) the initial solution will be the same: $j_1 = 1$, $j_2 = 2$, $j_3 = 3$, $j_4 = 4$; but chains (1,4) and (4,3), which were improving in the case of matrix (21) are not improving now because

$$c_{11} + c_{44} = c_{1j_1} + c_{4j_4} = 12 + 9 < c_{41} + c_{14} = 3 + 19.$$
$$c_{22} + c_{44} = c_{2j_2} + c_{4j_4} = 12 + 9 < c_{42} + c_{24} = 4 + 19.$$

There are no improving chains of length one, but there is an improving chain (4,1), (1,3) of length two:

$$c_{11} + c_{33} + c_{44} = c_{2j_2} + c_{3j_3} + c_{4j_4} = 12 + 10 + 9 > c_{14} + c_{43} + c_{31} = 19 + 5 + 6.$$

Using this improving chain, we get the solution $j_1 = 4$, $j_2 = 2$, $j_3 = 1$, $j_4 = 3$. Applying to this solution the improving chain (2,3), we get an optimal solution $j_1 = 4$, $j_2 = 1$, $j_3 = 2$, $j_4 = 3$.

COMPUTATIONAL RESULTS AND COMPARISON WITH THE NETWORK APPROACH

The algorithm was written in FORTRAN language for an IBM4341/12 using the CMS operation system. There are two versions of the algorithm that differ in the precision or the amount of digits needed for the presentation of the elements of the cost matrix. The first version is constrained by presenting costs as integers occupying not more than four bytes in binary presentation (i.e., as standard integers in FORTRAN). This algorithm has no constraints on the length of improving chains and minimal constraints in selection of these chains in the improving part of the algorithm. We used this program to compare the performance of the algorithm for the pure assignment problem. In all comparisons with commercial software package GNET performance for the same problems, we found the computational times were approximately the same. Dimensions of constraint matrices of explored problems were as large as 233. We will not spend time on this part of the comparison because the main purpose of the project was to develop tools for solving assignment problems with side constraints.

The second version of the algorithm was designed to compare the effectiveness of the algorithm with the network implementation of the Navy assignment problem for nonrated personnel (Seaman, Airman, Fireman, and Fireman Apprentice; this is described in the Model Formulation section of this report and in Liang & Thompson, 1986). This problem is computationally aggravated because the cost matrix for the problem has elements that are integers occupying up to 12 bytes in binary presentation. FORTRAN has no special mechanism for working with integers this large, so we were forced to use FLOAT16 arithmetic to work with these costs. Since FLOAT16 arithmetic is known to be very slow, we were forced to do different "tricks" to accelerate the speed of the algorithm. Along with the cost matrix, we used another matrix, elements of which are standard FORTRAN integers and equal to the order between different costs. This matrix was used for comparing elements of the cost matrix, because comparing FLOAT16 numbers is a very time-consuming process. In this version of the algorithm, we stopped after checking improving chains of length 2 in the improving part of the algorithm. This appeared to be quite adequate in all the practical examples. Of course, using improving chains of length not more than two converts this algorithm to a heuristic. For simplicity reasons we made all penalty coefficients in (18) equal (i.e. $q_1 = q_2 = q_3 = q_4 = q$), and we measured q with respect to $\max_{ij} c_{ij}$

where the maximum is based on the eligible combination (i,j). We found that increasing the penalty coefficient with respect to this maximum (Max) yields more computational time, because the amount of suspicious chains needing to be checked in the improving part of the algorithm also increases.

Computational results are presented for two cases in the following tables. The first, shown in Tables 1.1, 1.2, and 1.3, on page 17 describe a situation where 81 persons are assigned to 163 jobs. All jobs belong to one major group and two minor subgroups. Due to the manning balance constraint, the manning percentage in the ideal case should be the same in both subgroups and was reached by the algorithm with the penalty coefficient of 1/2 Max. Reaching this balance increases the cost of the assignment.

The second example, presented in Tables 2.1, 2.2, and 2.3 on page 18, exemplifies 72 persons who should be assigned to 145 jobs. The jobs belong to three different major groups. The first major group has two subgroups, the second has one subgroup, and the third has three subgroups. The second major group is constrained by the maximal manning percentage as in inequality (4). As in the first example, nearly ideal allocation is reached with the penalty coefficient equal to 1/2 Max; but this time, the cost of all assignments is better with the algorithm application than with the network code. Some differences in allocations with the network approach and the new algorithm solution can be partly explained by making all penalty coefficients in the objective function (8) equal.

GROUPS	Vacancies	People on Board	Manning Percentage
Group 1, Subgroup 1	5681	4184	73.7%
Group 1, Subgroup 2	5035	3660	72.7%

Table 1.1
Original Data (Example 1)

GROUPS	Allocation	Manning percentage
Group 1, Subgroup 1	17	73.9%
Group 1, Subgroup 2	64	74%
Cost	10013001440276	
CPU Time in Sec.	28.1	

Table 1.2
Network Approach (Example 1)

	1/200 Max		1/5 Max		1/2 Max	
GROUPS	Alloc.	Percent	Alloc.	Percent	Alloc.	Percent
Group 1, Subgroup 1	24	74.4%	17	73.9%	16	74%
Group 1, Subgroup 2	57	73.8%	64	74%	65	74%
Cost	10007751390055		10014401454977		10015451442724	
CPU Time in Sec.	29.3		35.6		38.2	

Table 1.3
Algorithm Results (Example 1)
Penalty Coefficients in Terms of Maximal Cost

GROUPS	Vacancies	People on Board	Manning Percentage
Group 1, Subgroup 1	5663	4184	73.9%
Group 1, Subgroup 2	5164	3784	73.3%
Group 2	159	157	98.7%
Group 3, Subgroup 1	699	309	44.2%
Group 3, Subgroup 2	1168	522	44.7%
Group 3, Subgroup 3	1381	609	44.1%

Table 2.1
Original Data (Example 2)

GROUPS	Allocation	Manning percentage
Group 1, Subgroup 1	20	74.2%
Group 1, Subgroup 2	31	73.9%
Group 2	2	100%
Group 3, Subgroup 1	5	44.9%
Group 3, Subgroup 2	2	44.9%
Group 3, Subgroup 3	12	45%
Cost	8170251272896	
CPU Time in Sec.	17.1	

Table 2.2
Network Approach (Example 2)

	1/200 Max		1/5 Max		1/2 Max	
GROUPS	Alloc.	Percent	Alloc.	Percent	Alloc.	Percent
Group 1, Subgroup 1	20	74.2%	7	74%	7	74%
Group 1, Subgroup 2	28	73.8%	37	74%	37	74%
Group 2	4	101.3%	4	101.3%	4	101.3%
Group 3, Subgroup 1	9	45.6%	10	45.6%	7	45.2%
Group 3, Subgroup 2	4	45%	4	45%	4	45%
Group 3, Subgroup 3	8	44.7%	10	44.8%	13	45%
Cost	7925251285973		7935751283523		7936451310473	
CPU Time in Sec.	16.9		21.5		24.3	

Table 2.3
Algorithm Results (Example 2)
Penalty Coefficients in Terms of Maximal Cost

CONCLUSIONS

The search-by-chain algorithm achieved an objective function value very close to results achieved by GNET. The algorithm used 5 percent to 30 percent (depending on value of penalty coefficients) more computational time than the GNET algorithm when applied to a few examples of the Enlisted Personnel Allocation and Nomination System (EPANS) model for nonrated personnel. Because the EPANS formulation is a partly heuristic process, it is reasonable to have two separate algorithms based on different approaches to check the quality of the solution. In addition, the search-by-chain algorithm is more flexible for incorporating additional side constraints to handle more complex Navy assignment problems.

FUTURE DEVELOPMENT

More extensive computational comparisons will be performed in the future on more complex Navy problems, particularly ones that contain multiple school quota constraints. In addition, the algorithm will be reprogrammed in more advanced computer languages, such as assembly and C, to make it more computationally efficient.

REFERENCES

Barr, R., Farhangian, K., & Kennington, J. F. (1986). Network with constraints: An LU factorization update. *The Annals of the Society of Logistics Engineers, 1,* 66-85.

Bazaraa, M.S., & Jarvis, J.J. (1977). *Linear programming and network flows.* New York: Wiley and Sons.

Blanco, T.A., Liang, T.T., Habel, G., & Ritter, F. (1984, May). *Automated enlisted personnel allocation process: Development and testing* (NPRDC Tech. Rep. 84-41). San Diego: Navy Personnel Research and Development Center. (AD-A141 953)

Bradley, G., Brown, G., & Graves, G. (1977). Design and implementation of large-scale primal transshipment algorithms. *Management Science, 24,* 1-35.

Charnes, A., & Cooper, W. (1961). *Management models and industrial applications of linear programming* (Vols. 1-2). New York: Wiley and Sons.

Chen, H., & Saigal, R. (1977). A primal algorithm for solving a capacitated network flow problem with additional linear constraints. *Networks 7,* 1.

Glover, F., Karney, D., & Klingman, D. (1974). Implementation and computational comparison of primal, dual, and primal-dual computer codes for minimum cost network problem. *Networks, 4,* 191-212.

Glover, F., Karney, D., Klingman, D., & Russel, R. (1978). Solving singly constrained transshipment problems. *Transportation Science, 12,* 277-297.

Glover, F., Klingman, D., & Ross, G.T. (1973a). *Finding equivalent transportation problems* (Res. Rep. CCS 107), Austin, TX: University of Texas, Center for Cybernetic Studies.

Glover, F., Klingman, D., & Ross, G.T. (1973b). *Finding equivalent network formulations for constrained network problems* (Res. Rep. CCS 108), Austin, TX: University of Texas, Center for Cybernetic Studies.

Karney, D., & Klingman, D. (1976). Implementation and computational study on an in-core out-of-core primal network code. *Operation Research, 24,* 1056-1077.

Kennington, J., & Helgason, D. (1980). *Algorithms for network programming,* New York: Wiley and Sons.

Kuhn, H.W. (1953). Hungarian method for assignment problem. *Naval Research Logistics Quarterly, 2,* 83-97.

Liang, T.T. (1984, July). *A network formulation of multiple criterion problems for developing an integrated personnel distribution system in the Navy* (NPRDC Tech. Rep. 84-49). San Diego: Navy Personnel Research and Development Center. (AD-A141 204)

Liang, T.T., & Lee S. (1985). A system approach to integrate manpower planning and operation. *Socio-Economic Planning Sciences, 19, (6),* 371-377.

Liang, T.T., & Thompson, T.J. (1986, July). *Optimizing personnel assignment in the Navy: The seamen, fireman and airman application* (NPRDC Tech. Rep. 86-24). San Diego: Navy Personnel Research and Development Center. (AD-A141 204)

Mulvey, J. (1978). Pivot strategies for primal simplex network codes. *Journal of the Association for Computing Machinery, 25,(2),* 266-270.

APPENDIX

PROOFS OF THEOREMS

Proof of Lemma 1

It is obvious that for the first person selected by the gradient approach, person i_1, we have the equality

$$c_{i_1 j_1} = \min_j c_{i_1 j},$$

where j_1 is the assignment to the person i_1 by the gradient approach. Then for the second person, person i_2, we will have the equalities

$$c_{i_2 j_2} = \min_{j \in \{1,\ldots,n\} - \{j_1\}} c_{i_2 j} = \min_j c_{i_2 j},$$

where j_2 is the assignment to the person i_2. The second equality is due to assumed monotonicity. Thus in both cases we have assigned persons to minimizing jobs. If we continue this consideration, by induction we will get Lemma 1.

Proof of Lemma 2

Let the initial solution of the gradient approach be $\{j_1,\ldots,j_k,\ldots,j_n\}$, where j_k is the job assigned to the person k. Without lost of generality, we can assume that person k is being assigned at stage k of the algorithm. Assume that assignment $\{j_1,\ldots,j_n\}$ is not optimal; that is, there is an assignment $\{\bar{j}_1, \ldots, \bar{j}_n\}$, the cost of which is less than that of assignment $\{j_1,\ldots,j_n\}$. Let r be the person, such that $\bar{j}_r = j_1$; then consider a new assignment:

$$\{j_1, \bar{j}_2, \bar{j}_3, \ldots, \bar{j}_{r-1}, \bar{j}_1, \ldots, \bar{j}_n\}. \tag{1A}$$

Assignment (1A) differs from the original assignment $\{\bar{j}_1, \ldots, \bar{j}_n\}$ by elementary permutation $(1,r)$. By definition of the gradient approach we have

$$c_{1 j_1} \leq c_{1 \bar{j}_1}.$$

If in addition to that we have

$$c_{r \bar{j}_1} \leq c_{r \bar{j}_r} = c_{r j_1}, \tag{2A}$$

we see that the cost of assignment (1A) is not more than the cost of $\{\bar{j}_1, \ldots, \bar{j}_n\}$.

If contrary to inequality (2A) we have:

$$c_{r\bar{j}_1} > c_{rj_1}, \tag{3A}$$

then due to the lemma conditions,

$$c_{r\bar{j}_1} - c_{rj_1} < c_{1\bar{j}_1} - c_{1j_1},$$

or

$$c_{r\bar{j}_1} + c_{1j_1} < c_{rj_1} + c_{1\bar{j}_1} = c_{r\bar{j}_r} + c_{1\bar{j}_1},$$

which also shows that assignment (1A) cost less than assignment $\{\bar{j}_1, \ldots, \bar{j}_n\}$. Repeating this process with elementary permutations not more than n times, we will get assignment $\{j_1,\ldots,j_n\}$ out of assignment $\{\bar{j}_1, \ldots, \bar{j}_n\}$. Because at every step of this process the cost of intermediate assignment is decreasing, we will finally contradict our assumption (3A), which proves the Lemma.

Proof of Lemma 4.

Let $\{\bar{j}_1, \ldots, \bar{j}_n\}$ be an assignment obtained from the original assignment $\{j_1, \ldots, j_n\}$ with the help of elementary permutation (r,k); that is, $\bar{j}_r = j_k$; $\bar{j}_k = j_r$ and $\bar{j}_i = j_i$ for $i \neq r$; $i \neq k$.

If contrary to the Lemma we have

$$c_{rj_r} \leq c_{rj_k} = c_{r\bar{j}_r},$$

then we should get

$$c_{kj_k} > c_{kj_r} = c_{k\bar{j}_k}. \tag{4A}$$

This is because in the other case, the cost of assignment $\{\bar{j}_1, \ldots, \bar{j}_n\}$ will not be less than the cost of the original assignment $\{j_1,\ldots,j_n\}$, which contradicts the assumption of the improving chain. But inequality (4A) differs from the Lemma's statement only in renumeration of persons, which proves the Lemma. Indeed, if we denote person k as person r and vice versa, statement (4A) will coincide with the Lemma statement.

DISTRIBUTION LIST

Deputy Under Secretary of Defense for Research and Engineering (Research and Advanced Technology)
Military Assistant for Training and Personnel Technology (OUSD) (R&AT)
Assistant for MPT Research and Development and Studies (OP-01B7)
Commanding Officer, Naval Training Systems Center
Director, Office of Naval Research (OCNR-10)
Head, Cognitive and Neuro Sciences (OCNR-1142)
Chief Scientist, Office of Naval Technology (OCNR-20T)
Technology Area Manager, Office of Naval Technology (Code 222)
Office of Naval Research, Detachment Pasadena
Office of Naval Research, London
Technical Director, U.S. ARI, Behavioral and Social Sciences, Alexandria, VA (PERI-ZT)
Commander, Air Force Human Resources Laboratory, Brooks Air Force Base, TX
Air Force Human Resources Laboratory, TSRL/Technical Library (FL 2870), Brooks Air Force Base, TX
Program Manager, Life Sciences Directorate, Bolling Air Force Base, DC (AFOSR/NL)
Commanding Officer, U.S. Coast Guard Research and Development Center, Avery Point, Groton, CT
Superintendent, Naval Postgraduate School
Director of Research, U.S. Naval Academy
Program Manager, Manpower Research and Advisory Service, Smithsonian Institute
Institute for Defense Analyses, Science and Technology Division
Center for Naval Analyses
Canadian Forces Personnel, Applied Research Unit, Canada
Ministry of Defense, Senior Psychologist, Naval, England
D. Dennison, Army Personnel Research Establishment, Personnel Psychological Division, England (2)
Science 3, RAF, Lacon House, England
Directorate of Psychology, AF, Department of Defense, Air for CE, Australia
Navy Psychology, Australia (2)
Defense Psychology Unit, Defense HQ, New Zealand (2)
Defense Technical Information Center (DTIC) (2)

DEPARTMENT OF THE NAVY

NAVY PERSONNEL RESEARCH AND DEVELOPMENT CENTER
(CODE___)
SAN DIEGO, CA 92152-6800

OFFICIAL BUSINESS
PENALTY FOR PRIVATE USE, $300

U23163I

U.S. OFFICIAL MAIL

U.S.MAIL

Postage and Fees Paid
Department of the Navy
DOD-316